멀티미디어 교실을 위한 윈도우 NT 기반 스트림 서버 구현

손주영[†]

요 약

개인화된 학습내용과 진도로 멀티미디어를 이용한 교재를 통해 학습 효과를 크게 제고할 수 있는 중등학교 멀티미디어 교실과 대학의 멀티미디어 센터를 위한 분산 스트림 서버 시스템을 구현하였다. 기존의 멀티미디어 정보 재생 시스템은 멀티미디어 교실에 적용하기에 적절하지 못한 제약점을 가지고 있다. 과다한 스트림당 비용이 요구되거나 그렇지 않으면 학습에 활용하기에는 저급한 재생 품질, 원활하지 못하는 시스템 및 서비스 확장성, 개별적 고유 클라이언트 환경에 의한 사용 이질감, 교사 조작 능력과 표현 의도가 전혀 고려되지 않은 일반적 저작 도구로 인한 교재 저작 어려움, 그리고 구성 시스템간의 유기적 연동 부재로 인한 관리 어려움 등의 문제점을 극복한 시스템을 구현하였다. 폐쇄되어 있는 교실에서뿐만 아니라 인터넷을 통한 광범위한 원격 교육에 확장할 수 있도록 웹 기반 분산 시스템으로 구성하였다. 전체 시스템의 구성 요소는 멀티미디어 정보 저장 및 재생을 담당하는 스트림 서버, 클라이언트 시스템, 분산되어 있는 서버의 통합 역할을 하는 서비스 게이트웨이, 그리고 클립 및 교재 저작을 위한 저작관리 시스템 등이다. 본 논문에서는 그 가운데 멀티미디어 정보를 저장, 전송하는 스트림 서버의 설계 및 구현에 대해 설명한다. 윈도우 NT 서버에서 실행되는 한 대의 스트림 서버 시스템으로 한 학급의 클라이언트(50~60대)에서 MPEG-1 스트림을 동시에 재생할 수 있는 성능을 아무런 시스템 변경 없이 응용 수준의 소프트웨어 엔진만으로 실현하였다. 그리고 타 구성 요소 시스템간의 유기적 연동을 통한 시스템의 확장성과 서비스의 유연성을 확보할 수 있었다.

Implementation of a Windows NT Based Stream Server for Multimedia School Systems

Jooyoung Son[†]

ABSTRACT

A distributed multimedia school system is developed for the multimedia classroom at high school and university. The system is designed and implemented for students to improve the learning efficiency through the personalized multimedia contents and pace of learning. The previously developed multimedia information retrieval systems have some limitations on being applied to the multimedia classroom: expensive cost per stream or poor retrieval quality inappropriate for education, unscalability of system and service, unfamiliar proprietary client environment, and difficulty for teachers to use the authoring tools and manage the authored teaching materials. The system we developed overcomes the above problems. It is so scalable as to be applicable not only to a segmented classroom but also to the world wide Internet. The stream server is one of the components of the system: stream servers, clients, a service gateway system, and a authoring management system. This paper describes the design and implementation of the stream server. A single stream server can simultaneously playback the multimedia streams as many as clients at one classroom. This is achieved only by the software engine without any changes of the hardware architecture. The systematic coupling with other components gives the scalability of the system and the flexibility of services.

본 연구는 LG전자(주)와 한국해양대학교 학술진흥회 학술연구지원사업의 지원에 따라 수행되었음.
[†] 한국해양대학교 자동화정보공학부

1. 서 론

현재 전국 일선 중·고등학교에 설치되고 있는 멀티미디어 교실은 기존의 교육방식을 크게 벗어난 새로운 교육 및 학습 환경을 구축하는 기반 시설로써 많은 기대를 받고 있다. 인쇄 매체에 의한 단일 교재를 통한 일률적 주입식 교육이, 사실 표현력이 훨씬 큰 멀티미디어 교재에 의한 학생 개개인의 차별화 및 개별화된 교육으로 이행하는 효과를 얻을 수 있다. 교사들의 창의적 교재 제작이 가능하게 되고 나아가 교육전산망(EDUNET)을 통해 학교, 교원간에 활발한 정보교류가 가능하게 됨으로써 교육적 효과가 크게 제고된다[1,2].

1.1 기존 연구개발 현황 및 문제점

기존의 멀티미디어 교실을 위한 시스템은 크게 교재 제작을 위한 각종 멀티미디어 기기와 교원용 PC, 제작된 교재를 저장·재생하는 스트림 서버 그리고 약 60대의 학생 PC로 구성된다(그림 1).

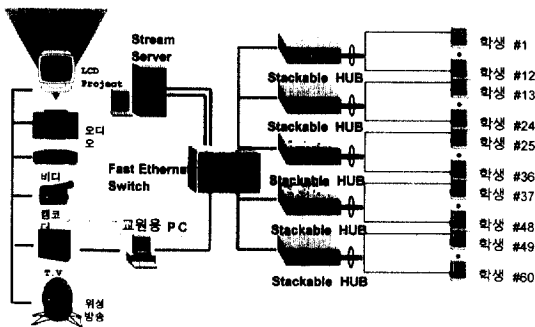



그림 1. 일반적인 멀티미디어 교실

이 가운데 가장 핵심적인 기술은 멀티미디어로 구성되는 교재를 저장, 재생하는 스트림 서버를 설계, 구현하는 것이다. 스트림 서버는 설계 및 구현 방식, 그리고 운영되는 환경에 따라 각각 두 부류로 나뉜다. 우선 설계 및 구현 방식에 의거하여 분류하면 다음과 같다. 첫째, 하드웨어의 구성요소만을 멀티미디어 데이터 처리를 위해 변경한 형태이다. 이 경우 서버 내에 소프트웨어로 구현되는 스트림 엔진 없이 하드웨어의 성능만으로 멀티미디어 데이터의 저장, 전송을 수행한다[11]. 주로 저장장치와 통신 인터페이스 카드의 전송률을 높이기 위해 고성능 ultra wide SCSI 인터페이스와 하드웨어 RAID, 그리고 100Mbps Fast Ethernet 또는 ATM 인터페이스를 채택한다. 그러나 이 방식은, 기존의 운영체제 및 응용 프로그램의 연속매체 재생에 대한 부적합성으로 하드웨어의 성능을 최종 응용 단계까지 보장해 주지 못하는 단점을 가진다[5,7].

둘째, 멀티미디어 연속매체 저장 및 전송을 위한 소프트웨어 엔진을 구현한 형태이다. 이 경우 하드웨어의 성능을 최대한 최종 사용자들에게 보장한다[12]. 엔진은 주로 저장장치의 연속매체에 대한 읽기, 쓰기 알고리즘, 스케줄러를 구현하여 단위 시간당 읽는 량을 최대로 하고, 연속매체를 위한 전송 메커니즘, 프로토콜을 수용함으로써 전송률을 최대로 유지하는 기능을 가진다[5,6,7,8]. 일반적으로 알고리즘과 프로토콜은 시스템 커널 스레드, 또는 디바이스 드라이버 형태로 구현된다[13]. 이 방식에 의해 구현된 시스템은, 일반 문서와 멀티미디어 연속매체 파일의 처리를 분리하여야 하고, 후방 호환성을 유지하기 위한 파일 시스템과 통신 관련 API, system call, 그리고 utility등이 모두 새롭게 작성되어 사용자에게 제공되어야 하는 문제를 가진다. 그리고 운영체제의 업그레이드에 맞추어 기존의 엔진을 수정하거나 완전하게 새롭게 작성되어야 한다. 단일 시스템에서 100 사용자 이상의 동시 재생 서비스가 가능해야 하는 대규모의 스트림 서버인 경우에는 적합할 수 있지만 멀티미디어 교실과 같은 중소 규모(50~60 동시 사용자)의 서버만을 필요로 하는 환경에서는 비용 대 성능 면에서 바람직하지 않다.

운영 환경에 따라 분류하면 다음과 같이 나뉜다. 첫째, 인터넷 웹 환경에서 운영되는 시스템이다. 인터넷의 전송지연가변성(jitter)이 매우 크고, 각 채널에 할당되는 대역폭이 매우 적은 환경적 제약으로 인해 재생 품질이 열악한 편이다. 동화상 경우, 재생율과 화면 크기 등에서 화상 채팅과 인터넷 방송 등의 응용에 사용할 수 있을 정도이다. 동화상의 압축 알고리즘은 H.263을 비롯하여 이와 유사한 화질을 내는 개발 회사 고유의 알고리즘(AVI, MOV, RM 등)을 이용한다. 이에 속하는 예를 보면 MicroSoft사의 NetShow, Real Network사의 Real Video, Vivo Software사의 Vivo Active, 그리고 Macromedia사

의 Shockwave 등이다. 이들 제품은 일반 사람들에게 가장 친숙한 인터넷 환경인 웹에서 제공되기 때문에 사용자 친숙성 면에서는 좋은 특성을 가지고 있다. 그러나 좋지 않은 품질로 재생되는 동화상을 통해 교육적 효과를 얻기에는 부적합하다.

둘째, 높은 대역폭, 낮은 지터 등의 특성을 제공하는 인트라넷 환경에서 운용되는 시스템이다. 통신망의 성능으로 인한 제약점이 해소됨에 따라 고품질의 재생 서비스가 가능하다. 따라서 MPEG과 같은 알고리즘으로 인코딩되어 있는 동화상 정보가 실시간적으로 전송(streaming)되면서 재생된다. 또한 재생 도중에 VCR 제어 기능과 같은 강력한 사용자 상호작용도 허용되고 있다. 이에 속하는 것들은 Sun사의 MediaCenter, Starlight Networks사의 StarWorks, HP의 MediaStream Server, Xing사의 StreamWorks, VXTREME사의 Web-theater, 그리고 SGI사의 WebFORCE MediaBase 등이다. 이들 시스템들은 주로 자사 고유의 하드웨어/운영체제 플랫폼을 바탕으로 구성된다. 그러나 설치되는 장비의 범용성이 떨어져 특정 응용(VOD)에만 사용할 수밖에 없다. 확장성도 보장되지 않고 있다. 또한 사용자 환경 측면으로 보았을 때 고유의 클라이언트 프로그램을 설치, 사용하여야 하므로 인터넷을 접근할 때 일반적으로 사용하는 웹 패러다임과 다르기 때문에 익숙하지 않다. 특히 이들 시스템은 대규모 사용자 집단을 목표로 하기 때문에 소규모 사용자를 상대로 하는 멀티미디어 교실에 적용하기에는 스트림당 비용이 매우 크다. 또한 응용을 다양하게 작성할 수 있는 도구(API) 등이 부족하며, 교재 저작 도구도 타 회사 제품으로 별도로 제공되고 있기 때문에 일선 교사들의 조작 능력에 맞지 않아 교재 저작이 매우 어렵다. 그리고 저장되는 교재와 교재에 포함되는 멀티미디어 정보에 대한 통합적 관리가 어려운 단점을 가진다.

기존 시스템에 대한 분석을 고려하면, 멀티미디어 교실에서 요구하는 품질과 확장성 그리고 응용의 유연성 등을 만족시키는 새로운 시스템의 개발이 절실함을 알 수 있다. 그러나 현재 국내에서 공급되는 멀티미디어 교실 시스템은 위의 문제점들을 가지고 있는 기존의 시스템이 주종을 이루고 있다. 대표적인 시스템들은 TNCI사의 Cheetah, ATML사의 VS 4000, Starlight Networks사의 StarWorks, 그리고 Xing사의 StreamWorks 등이다.

1.2 본 시스템의 특성

분산 멀티미디어 시스템의 요구사항 - 이러한 기존 시스템의 제약점들을 극복하는 멀티미디어 교실을 위한 분산 멀티미디어 시스템을 새롭게 설계, 구현하였다. 시스템의 요구사항을 요약하면 다음과 같다. 학생용 클라이언트 시스템의 사용자 인터페이스는 친숙한 웹으로 한다. 단일 스트림 서버가 지원하는 최대 동시재생 스트림 수는 하나의 교실에 있는 모든 클라이언트를 커버한다. 재생 품질은 MPEG-1을 포함하여 H.263 등 넓은 스펙트럼을 지원한다. 이로 인해 인트라넷에서 인터넷으로 응용의 유연성을 확보한다. 동시에 동일한 스트림이 여럿 재생될 때 통신망과 스트림 서버의 부하를 경감시키는 멀티캐스팅이 가능하다. 여러 스트림 서버를 둘 수 있고, 각 서버에 걸리는 부하를 균등하게 분할함으로써 전체 시스템 성능을 극대화한다. 이것으로 전체 시스템의 확장성을 보장한다. 마지막으로, 교사들이 조작하기 쉬우면서도 교사의 교육 의도를 충분히 반영할 수 있는 사용자 인터페이스를 가지고 있는 교재 저작도구가 필수적이다. 교재 제작을 위해 필요한 클립 정보 수집, 교재 저작, 그리고 저작된 교재의 등록, 저장, 수정, 삭제 등의 일련의 작업을 통합적으로 할 수 있는 저작관리 환경을 제공한다.

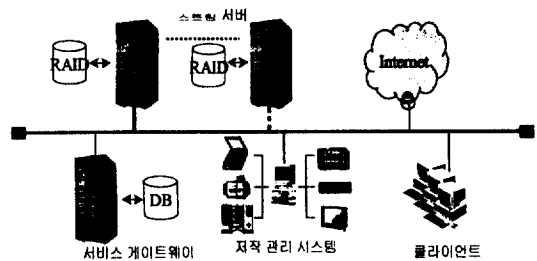



그림 2. 분산 멀티미디어 교실 시스템 구조

분산 멀티미디어 시스템 구조 - 멀티미디어 교실에 적용하기 위해 설계, 구현된 분산 멀티미디어 시스템은 멀티미디어 자료를 저장, 재생하는 스트림 서버 및 클라이언트 시스템, 교재의 등록, 저장 및 클라이언트 재생 요청 제어 기능을 담당하는 서비스 게이트웨이, 그리고 교재 제작을 위한 저작 관리 시스템 등으로 구성되어 있다[3]. 스트림 서버, 서비스 게이트웨이, 그리고 저작 관리 시스템은 고유 기능을 담

당하도록 논리적으로 완전히 분리된 시스템으로 설계, 구현되었다. 그러나 반드시 물리적으로 각각 독립적인 컴퓨터 시스템에 설치, 운영되어야 하는 것은 아니다. 구축된 하드웨어 환경에 따라 물리적 시스템과 여러 가지 결합 형태를 가질 수 있다. 예를 들어, 그림 2.는 컴퓨터와 1 대 1 결합 형태로 구성된 시스템 구조를 보여 준다. 이러한 실행환경의 유연성은 개별적인 교실 내의 시스템 구성 상황에 따라 적절하게 적용할 수 있게 하는 특징을 나타낸다.

스트림 서버는 멀티미디어 클립(제작된 교재에 포함되어 있는 오디오, 비디오 정보)을 저장하고, 클라이언트의 재생 요청을 받으면 수용 여부를 결정한 후 재생되는 클립의 재생 품질(재생율)에 맞추어 전송한다. 이를 위해 다양한 재생율을 지원하는 실시간 스트리밍 전송 기술을 구현하였다[5,6,7,8]. 각 스트림 서버는 고속 이더넷 스위치(fast Ethernet switch)에 의해 100 Mbps의 전송 대역폭을 보장받는다. 동일한 자료를 많은 학생들이 동시에 학습하는 경우를 대비하여 시스템 및 통신망 자원 효율을 최대로 하기 위해 IP 멀티캐스팅 기술을 적용하였다[9]. 그리고 분산되어 있는 스트림 서버들이 서비스 게이트웨이를 통해 통합적인 서비스를 제공할 수 있게 되었다[10].

클라이언트 시스템은 하드웨어 또는 소프트웨어적으로 다양한 인지매체를 재생할 수 있는 멀티미디어 PC 이다. 현재 가장 친숙한 통신 환경인 웹에서 학습이 가능하도록 구현되었다. 클라이언트들은 IP 스위치 허브를 통해 각각 전용 10 Mbps의 전송 대역폭을 활용할 수 있어 고품질의 MPEG-2 동화상을 재생하는 것도 가능하다. 단가 절감, 이식성 제고를 위해 동화상 재생을 소프트웨어로 가능하게 하였다. 학생들은 웹 브라우저를 통해 서비스 게이트웨이에 접속하여 학습 홈페이지를 열고 학습할 내용을 선택한다. 학습 교재에 연속매체가 포함되어 있을 경우, 연속매체 플레이어가 작동된다. 이것이 스트림 서버로부터 연속 매체를 재생율에 따라 전송 받는 즉시 재생한다. 다른 매체 정보(발표자료, 정지화상 등)는 서비스 게이트웨이로부터 전송된다. 이는 스트림 서버가 연속매체의 저장, 전송만을 전담함으로써 보다 많은 스트림을 수용하면서 균일한 재생품질을 보장하는 특징적 메커니즘이다.

교사가 제작 또는 수집한 교재 또는 자료를 저장할 때 직접 스트림 서버에 접속, 저장하는 것은 자료 관리 어려움을 초래한다. 해결책으로 서비스 게이트웨이는, 함께 개발된 고유의 저작 도구와 연동하여 저작 과정에서 필요한 멀티미디어 클립에 관한 메타 정보를 제공하고, 완성된 교재(웹 문서)를 등록, 저장, 관리하는 기본 기능을 가진다. 스트림 서버내의 연속매체 정보 저장, 수정, 삭제, 이동 등의 관리 기능도 함께 담당한다. 이 시스템은 자료 관리 기능 이외에 웹 서버 역할도 함께 한다. 사용자들에게 서비스를 제공하는 사용자 인터페이스 역할을 동시에 수행하는 것이다. 더 나아가 여러 스트림 서버들이 통신망(예: EDUNET)에 분산되어 있는 환경에서는 통합 학습 서비스를 제공하고, 클라이언트의 재생 요청을 특정 서버와 연결시켜 주는 기능을 담당한다. 그리고 스트림 서버간의 부하를 균등하게 배분하는 기능도 구현하였다. 스트림 서버 측면에서는 자신의 위치 및 정보 관리 변경에 대한 사실을 모든 사용자에게 알릴 필요가 없게 된다. 따라서 고성능, 확장성, 위치 투명성 등을 확보하는 역할을 함으로써 학생들에게 더욱 다양하면서도 광범위한 교육의 기회를 제공할 수 있는 기반이 된다. 웹 서버를 서비스 게이트웨이로 활용하는 것은 서비스 투명성과 웹을 통한 서비스 확장성을 동시에 얻게되는 장점을 가지게 한다.

마지막으로 교사들이 쉽고 효과적으로 멀티미디어 교재를 제작할 수 있도록 하는 저작 도구로서 저작관리 시스템을 구현하였다. 시각적 사용자 인터페이스를 제공하여 오디오, 비디오 같은 연속매체 자료, 웹 문서 또는 발표 자료 등의 재생 관련 공간적, 시간적 동기화 정보를 지정할 수 있다. 기존 자료를 찾을 수 있는 검색 기능도 포함된다. 미리보기를 통해 오디오, 비디오 자료를 편집할 수 있으며, 재생시의 VCR 조작기능을 위한 메타 정보도 만들어 낸다. 교사들의 조작 수준을 고려하여 매우 직관적이면서 쉬운 사용자 인터페이스(템플릿 기반 공간 지정, 시간축 기반 시간 동기화 지정 등)를 통해 편리하게 교재 저작을 할 수 있다. 최종 결과물은 서비스 게이트웨이가 저장, 관리하는 HTML 웹 문서이다.

본 논문은, 멀티미디어 교실을 위한 분산 멀티미디어 시스템의 구성요소 가운데 멀티미디어 자료의 저장, 전송을 담당하는 스트림 서버 시스템의 설계 및 구현에 관한 것이다. 스트림 서버는 하드웨어의 성능을 최대한 최종 사용자에게 보장하면서 동시에

무거운 소프트웨어 엔진의 가격 대 성능, 및 유지보수의 문제점을 개선하면서 다양한 서비스 품질을 동시에 제공할 수 있는 특성에 초점을 맞추었다. 따라서 커널 스레드나 디바이스 드라이버로 엔진을 구현하지 않고 응용 수준에서 엔진을 구현하였다. 이는 기존의 응용체제를 그대로 이용하는 반면 운영 환경에서 실시간 우선순위 프로세스로 실행시키고, 각 스레드의 주기적 실행을 보장함으로써 위의 요구사항을 만족시킬 수 있는 스트림 서버를 구현할 수 있었다[4]. Windows NT 4.0 서버 시스템 플랫폼에서 Win32 console program으로 응용 수준에서 구현된 스트림 서버에 관해 논한다.

논문의 구성은 다음과 같다. 2 장에서는 스트림 서버 구조와 서비스 시나리오를 설명한다. 스트림 서버 구현에 있어서 가장 어려운 기술적 논제를 중심으로 설명한다. 먼저 스트림 데이터를 읽고 전송하는 기능의 구현은 3 장에서 언급된다. 서비스 게이트웨이, 클라이언트와의 연동을 통한 세션 제어 기술에 대한 것을 4 장에서 설명한다. 5 장에서는 일반적인 학습교재에 많이 나타나는 서로 다른 매체간의 시공간적 동기화 처리 기술을 보이고, 6 장에서는 구현된 스트림 서버를 활용한 클라이언트의 실행 예를 소개한다.

2. 스트림 서버 구조 및 서비스 시나리오

스트림 서버 구조 - 클라이언트로부터 수신한 스트림 재생 서비스 요청에 따라 실시간 스트리밍 전송 서비스를 한다. 이 때 하드웨어(저장장치, 통신망 인터페이스 카드)가 내는 최고 전송 성능을 최종 서비스 단계에서도 그대로 발현시켜 각 스트림의 재생 품질을 유지하면서 동시 재생 스트림의 수가 최대가 되도록 하는 것이 스트림 서버의 핵심 기능이다. 그림 3.은 구현된 스트림 서버의 구조를 나타낸다. 여기서 굵은 화살표는 연속매체 데이터의 흐름, 가는 화살표는 제어 신호의 흐름을 나타낸다. 서버를 이루는 모듈 가운데 gateway handler는 분산 환경에 대비하여 게이트웨이와의 연동을 담당한다. 이로 인해 세션 시작 요청이 클라이언트와 스트림 서버 사이에 세션이 실제 열리기 전에 스트림 서버는 사전에 자원을 할당 가능성을 점검, 할당할 수 있고, 스트림 데이터를 미리 읽는다. 이러한 메커니즘은 시스템 전체적으로 보면 게이트웨이가 분산되어 있는 여러 스트림 서버 가운데 서비스 제공이 가장 유리한 서버를 선택할 수 있게 함으로써 시스템 전체적인 부하 균할을 기할 수 있다. 그리고 서버의 디스크로부터 데이터의 일부를 선반입함으로써 재생시작지연시간을 줄일 수 있는 특징을 가진다.

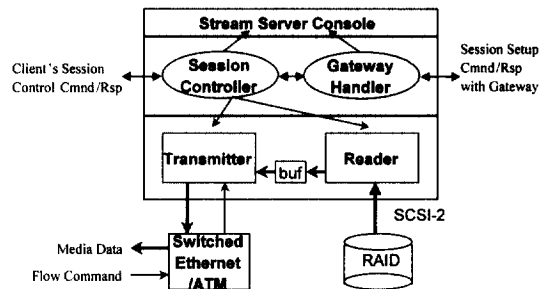



그림 3. 스트림 서버 구조

스트림 서버의 태스크들은 각각 능동적 태스크와 수동적 태스크로 나뉜다. 능동적 태스크는 외부의 명시적인 제어 신호를 받지 않더라도 일정한 시간 간격을 두고 주기적으로 기능을 수행하는 태스크이다. 여기에 속하는 것은 저장장치로부터 연속매체 데이터를 읽어오는 reader와 읽은 것을 통신망에 전송하는 transmitter 태스크이다. 이들은 재생되는 연속매체의 재생품질에 맞추어 단위시간 당 재생되어야 하는 데이터 량을 읽어 전송하는 일을 주기적으로 자발적인 방법으로 수행한다. 능동적 태스크로 설계한 이유는, 매 주기마다 연속매체를 읽고 전송할 때 클라이언트의 전송 요구를 받아서 전송하는 클라이언트-풀(client-pull) 방식으로 하는 것이 아니라 서버가 주기적으로 재생율에 맞게 데이터를 자발적으로 전송하는 서버-푸쉬(server-push)방식으로 스트림 서버를 설계하였기 때문이다. 서버-푸쉬 방식을 채택한 것은 하나의 세션을 통해 업-스트림(클라이언트로부터 서버로 전송되는 세션 제어 신호)과 다운-스트림(멀티미디어 연속매체 데이터)을 모두 송수신하는 상황에서는 업-스트림의 발생 빈도 및 크기를 최소로 하여 다운-스트림의 전송 효율을 최대화할 필요가 있기 때문이다.

수동적 태스크는 클라이언트, 서비스 게이트웨이, 또는 내부 다른 태스크로부터의 요청 신호가 있는 경우에 한해 해당 기능을 수행하고 그 결과를 회신한

다. 여기에 속하는 것은 서비스 게이트웨이의 세션 개시 신호에 반응하는 gateway handler, 세션이 열려 있을 때 수신되는 클라이언트의 세션제어 신호를 처리하는 session controller, 그리고 스트림 서버의 운행 상태를 감시 추적할 수 있도록 하는 stream server console 등이다.

이렇게 기능에 따라 능동 또는 수동적 태스크로 분리하여 구현한 것은, 운영체제의 round robin 방식 실행기회 부여로부터 탈피하여 서버의 성능에 가장 민감한 모듈인 reader와 transmitter에게 실행기회를 최대한 주기 위한 것이다. 멀티태스크 프로그램의 성능에 영향을 미치는 task switch의 부담을 감소시키기 때문이다[4].

서비스 시나리오 - 스트림 서버를 포함한 멀티미디어 교실 시스템의 전체적인 서비스 시나리오를 알아본다(그림 4). 서비스는 크게 세 단계로 구성되어 있다.

첫 단계는 서비스 세션을 구축하는 단계(service session setup phase)이다. 이 단계에서 클라이언트와 서비스 게이트웨이 그리고 스트림 서버의 연동이 요구된다. (1)먼저 클라이언트 시스템에서 학생은 일반적인 웹 브라우저를 사용하여 서비스 게이트웨이에 있는 학습 페이지에 접속한 후 학습하고자 하는 내용을 선택한다. (2)서비스 게이트웨이에서는 선택된 내용에 연속매체 정보가 포함되어 있는 경우 이 정보에 대한 메타정보(이름, 재생율, 재생위치 등)를 DB(SQL server)를 통해 검색한 후 전송, 재생 서비스를 할 스트림 서버를 분산되어 있는 여러 스트림 서버들 가운데 부하 균등화 원칙에 의거하여 선정한다. 선정된 서버에게 클라이언트의 IP 주소와 메타정보를 제어 명령(gateCommand)으로 전송한다. (3)스트림 서버내의 gateway handler가 명령을 수신하여 요청 받은 연속매체 데이터가 전송 가능한 것인지를 현재의 버퍼 및 저장장치, 통신망 대역폭 등의 자원 활용도를 보고 판단한다(서비스 수용 제어). (4)가능한 경우, 파일을 열고, 세션 제어 블록(SCB: session control block)을 할당하고 세션 id를 배정한다. (5)배정 받은 세션 id를 서비스 게이트웨이에게 제어 명령에 대한 응답(gateResponse)으로 회신한다. (6)서비스 게이트웨이는 성공적으로 스트림 서버가 세션을 열 수 있음을 클라이언트에게 선정된 스트림 서버의 IP 주소와 세션 id, 전송될 연속매체 데이터의 MIME 타입 등을 송신함으로써 클라이언트(웹 브라우저)의 재생요청에 대한 응답을 한다. 이때 연속매체와 함께 보여져야 하는 웹 페이지, 이미지, 발표자료(PPT 파일 등)들도 전송된다. 이로써 서비스 게이트웨이의 이 단계에서의 역할은 완료된다. (7)클라이언트(웹 브라우저)는 전송될 연속매체의 MIME 타입에 맞는 Plug-in program을 기동시킨다. 클라이언트(이때부터는 플러그인 프로그램)는 수신된 세션 id를 가지고 '세션 오픈'을 수신된 스트림 서버의 IP 주소로 요청한다. (8)스트림 서버내의 세션 관리를 담당하는 session controller가 세션 오픈 요청을 받아 세션을 연다. 이때 세션 id를 바탕으로 찾은 SCB 내에 담겨 있는 클라이언트의 IP 주소와 세션 요청한 것과 비교하여 사용자 인증을 한다. (9)확인이 완료되면, 클라이언트에게 ACK 회신을 보낸다.

둘째 단계는 형성된 세션에서 연속매체 데이터가 전송되고, 전송 흐름에 대한 제어를 하는 단계(streaming control phase)이다. 스트림 서버내의 reader와 transmitter는 클라이언트의 재생(play)요청이 오면 재생율에 맞는 데이터 량을 주기적으로 읽어 전송한다. 이때 클라이언트의 버퍼링 정도에 따라 흐름 명령(flow command)을 통해 전송 주기를 조절한다. (10)클라이언트는 열린 세션을 통해 재생(play), 일시 정지(pause), 되감기(rewind), 앞으로 가기(fast forward), 빠른 재생(fast play), 느린 재생(slow play), 정지(stop), 종료(end) 등의 연속매체에 대한 재생 속도 및 위치 제어(VCR control)를 할 수 있다. (11)이런 제어신호들은 스트림 서버내의 session controller가 수신하여 reader와 transmitter의 동작을 제어함으로써 VCR 제어를 실현한다.

셋째 단계는 세션을 닫는 단계(session teardown phase)이다. 재생을 끝내고 클라이언트와 스트림 서버사이의 세션을 없애는 단계이다. (12)클라이언트가 종료 제어신호를 서버에게 보낸다. (13)서버 내의 session controller가 이를 수신하여 부드러운 세션 종료(graceful session close) 작업을 시행한다. reader와 transmitter에게 작업종료를 알린다. 그들로부터 작업완료 신호가 오면 할당된 세션관련 자원을 되돌린다. 클라이언트에게 세션이 성공적으로 종료하였음을 알린다. 부드러운 세션 종료에 대한 것은 4 장에서 자세하게 설명한다.

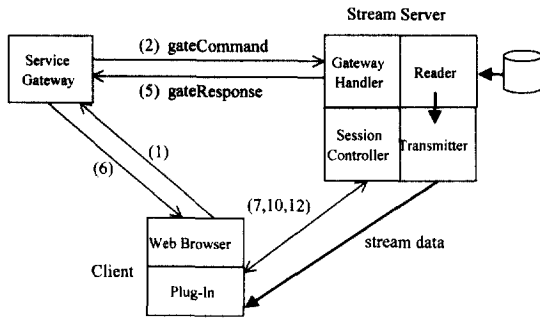



그림 4. 서비스 시나리오

3. 스트림 읽기와 전송

스트림 데이터 처리 태스크 구현 기법 - 연속매체 데이터를 저장장치로부터 읽고, 그것을 통신 인터페이스 카드로 전송하는 일로서 스트림 서버의 가장 핵심적 기능이다. 능동적 태스크로 설계된 reader와 transmitter는 스레드(thread)로 각각 구현되었다. 윈도우 NT의 선점식 멀티스레드 실행 환경(MTP)을 최대한 이용한 것이다. 이들은 주기적으로 읽고 전송하는 것을 수행하는 태스크들이므로 각각 자신의 실행시작시점을 정확하게 알 필요가 있다.

이를 보장해주는 기법으로 첫째 시스템 타이머 틱 인터럽트를 활용하는 방법(SetTimer()에 의해 발생하는 WM_TIMER 메시지)과 분주 대기(busy waiting)를 하면서 실시간을 읽어(GetTickCount()) 알아내는 방법이 있다. 그러나 실험 결과, 한 프로세스 내에서 타이머를 여러 태스크(스레드)가 공유하거나 2 개의 타이머를 각각 유지하여 개별적인 인터럽트를 활용하는 두 경우 모두 실제 실행시작시점의 알림 주기가 다른 태스크의 실행 지속 시간에 의해 영향을 받아 변한다는 사실을 발견하였다. 한 태스크의 작업량이 매 주기의 약 80%에 이르자, 다른 태스크의 타이머 인터럽트 주기가 매우 불규칙적으로 발생하는 현상이 일어남을 알게 되었다. 물론 두 개의 태스크에 작업량을 그 정도로 늘렸을 때는 두 태스크에서 모두 불규칙적인 타이머 인터럽트가 발생하였다. 또한 많은 세션이 동시에 실행될 때 각 세션별로 설정되는 실행시작시점을 정확하게 얻는 것은 이 방법으로는 불가능함을 알 수 있다. 따라서 타이머 인터럽트를 활용한 주기적 읽기와 전송은 그 정확성, 확장성을 확신할 수 없음을 알 수 있다. 이런 결과에 ***태스크 스케줄링*** - reader와 transmitter 관계는 생산자와 소비자 관계이다. 이때 생산자는 자신이 생산할 수 있는 최대의 성능으로 생산을 할 수 있도록 한다. 그러나 소비자의 소비속도에 맞추어 생산할 수밖에 없는 상황을 만든다. 그 방법은 생산자와 소비자 사이에 한정된 공유 자원을 두는 것이다. 즉, 읽은 스트림 데이터를 담을 수 있는 버퍼를 b 개로 한정한다. 이때 만약 b 개가 모두 차있다면, 생산자인 reader는 더 이상 버퍼가 없기 때문에 읽는 작업을 적어도 한 버퍼가 빌 때까지 일시 중단할 수밖에 없는 것이다. 따라서 만약 transmitter가 정확한 주기를 지키면서 소비한다면 reader 내부에서 별도의 읽기시작시점에 대한 고려는 필요 없다는 결론을 내릴 수 있다. 따라서 이것은 이중적 성격을 가진 스레드로 구성되는 프로세스에서 스레드를 스케줄링할 때 여러 데드라인을 두는 방식[17]에서 사실상 하나의 데드라인(여기서는 transmitter의 것)에 의해 다른 스레드의 실행 시간이 수동적으로 결정되는 방식으로 변형 설계된 방식이다. 프로세스 내부의 타이머 유지를 최소로 할 수 있음으로써 단순한 구현과 짧은 실행 시간 복잡도를 얻을 수 있다. 최종적으로 읽기와 전송의 정확한 주기성 보장은 transmitter 내부에서 분주대기 방식으로 계산되는 각 세션별 실행시작시점을 기준으로 구현되었다. 이점은 일반적인 주기적 스레드 스케줄 방식과 차이를 나타내는 특징을 가진다[4]. 시간 흐름에 대한 제어를 응용 내부에서 함으로써 좀더 시스템 내부의 지연에 의한 시간 부정확성의 문제를 회피할 수 있었다. 각 세션의 실행시작시점을 검색하여 가까운 시각에 실행을 해야 하는 순서로 읽기와 전송이 이루어진다(Earliest Deadline First). 실행시간 복잡도를 최소화하면서 성능을 얻을 수 있는 방법으로 EDF 알고리즘을 채택하였다. 위 프로그램 코드의 복잡도를 낮추는 것은 스케줄러를 가볍게 하여 스트림 서버의 성능에 가장 많은 영향을 주는 부분인 실제 읽고 전송하는 코드의 실행 기회를 늘리는 목적을 가진다.

전송 주기 및 블록 크기 결정 기법 - 한 주기동안 읽어서 전송하는 데이터 크기를 결정하는 방식은 주기를 모든 스트림에 대해 일정하게 고정하고 각 스트림의 재생율에 비례하여 결정되는 CTL(constant time length)와 임의의 일정 크기를 모든 스트림에

적용하여 주기를 재생율에 맞추어 조절하는 CDL (constant data length)방식 등이 있다[14]. 스트림 서버는 각기 다른 비트율(재생율)을 가지는 연속매체의 동시 재생도 가능할 뿐만 아니라 고정비트율(CBR)로 인코딩된 비디오 스트림뿐만 아니라 가변비트율(VBR) 비디오 스트림도 수용하도록 설계되었다. 따라서 위 두 방법의 혼합 형태인, 바로 직전에 전송한 데이터 량과 재생율에 의거하여 다음 읽어야 하는 시점과 데이터 크기도 동시에 결정하는 방법을 채택하였다.

이 방식을 통해 다양한 재생 품질과 재생율을 가지는 여러 멀티미디어 정보를 동시에 재생할 수 있는 커다란 장점을 얻을 수 있다.

VCR 제어 기능 처리 - 전송주기와 크기는 스트림 서버에서 지원되는 VCR 제어 동작과도 깊은 관련을 가지고 있다. MPEG 비디오 재생 때 VCR 제어를 지원하기 위해서는 다음 재생을 시작하는 시점은 항상 GOP의 시작시점이어야 한다[15]. 따라서 한 주기에 전송되는 단위를 한 GOP로 하면, client에서 VCR 조작 후 새롭게 재생을 재개하는 데이터의 시작시점을 수신한 데이터 가운데서 찾을 필요 없이 데이터의 처음부터 재생할 수 있게 되어 재생을 신속하게 재개할 수 있다. 그런데 CBR 비디오 경우 각 GOP 크기는 일정하다. 이 경우에는 한 스트림의 재생주기는 항상 일정하게 된다. 반면 VBR 경우에는 가변적이다. 따라서 이때는 매 주기마다 다음 주기에서 읽어야 하는 크기를 결정할 뿐만 아니라 그 크기에 따라 다음 주기의 시작시점(읽기시작시점)도 함께 계산하여야 한다. 한 주기동안 한 GOP를 전송할 때 하나의 패킷으로 보내는 것이 아니라 고정크기의 여러 패킷으로 분리하여 전송한다.

이렇게 VCR 조작을 원활하게 하기 위해서는 MPEG 파일 내에서 각 GOP의 offset 값을 사전에 미리 분석하여 알고 있어야 한다. 분석작업은 연속매체 클립에 대한 전처리 과정을 통해 저작관리 시스템에서 이루어진다. 스트림 서버는 GOP의 offset 값의 리스트를 오프라인 상태에서 저작관리 시스템으로부터 받아서 메타정보로 활용한다.

재생시작시간 단축 - 세션의 성립은 서비스 게이트웨이의 요청에 의해 서버내의 자원이 할당됨으로써 시작된다. 그리고 실제적인 세션은 클라이언트의 요청으로 성립된다. 이 둘간의 시간 차이를 활용하여 실제적인 클라이언트의 재생(play) 요청이 오기 전에 재생하여야 하는 스트림의 첫 두개의 GOP를 미리 읽어 캐시해 둔다. 이는 클라이언트의 재생 요청 후 실제 스트림의 재생이 시작되기까지의 지연 시간(playback startup delay time)을 크게 줄이는 역할을 하고 있다.

전송 프로토콜 - 서버의 운영 환경이 인터넷을 기반으로 하기 때문에 UDP/IP를 활용하였다. 인트라넷 환경에서 이루어진 전송 중에 소실되는 패킷으로 인한 품질 저하는 사람들이 인식하지 못할 정도로 미미하였다. 현재 개발된 RTP/RTCP 모듈을 DLL(RTP.DLL)로 작성하여 적용하는 중이다[16]. 여러 사람이 동시에 동일한 교재를 공부하는 상황에 대비한 방송형 서비스를 제공하기 위해 IP multicasting 프로토콜을 DLL(MUL.DLL)로 구현하여 적용하였다.

스트림 서버 태스크 모듈과의 연동은 DLL이 제공하는 API를 통해 이루어진다. 이것은 향후 RTP/RTCP와도 연동할 수 있도록 설계, 구현되었다. 멀티캐스트 세션 관리를 DLL에서 함으로써 스트림 서버의 세션에 관련된 부담을 크게 줄였다. 멀티캐스트 주소 그룹을 관리하는 것은 별도의 멀티캐스트 서버가 담당하게 하였다. 그림 5.는 구현된 프로토콜 스택과 구현된 프로그램 구조를 비교해서 보여준다.

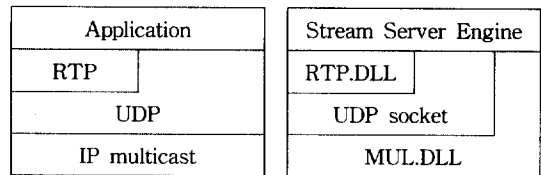



그림 5. 프로토콜 스택 및 구현 모델

4. 세션 제어

세션 개시 - 세션의 개시는 서비스 게이트웨이의 요청에 의해 자원활용도에 따라 서비스 가능 여부를 판단(service admission control)함으로써 이루어진다. 이때 서버내의 세션 제어 블록과 데이터 버퍼를 할당하고, 클라이언트의 실제적인 세션 오픈 요청에 대비하여 미리 데이터를 읽어오는(prefetch) 작업도 수행한다. 분산 서버 환경에 대비한 게이트웨이의 부하균등 기능과 재생시작지연시간을 줄이는 효과를 얻는다.

재생 위치, 속도 조절 - 클라이언트와의 세션이 열렸을 때 클라이언트로부터 오는 VCR 제어 신호에 따른 재생 위치와 속도조절 기능을 수행한다. 재생위치 조절은 RW, FF 등과 같이 현재 재생되고 있는 위치에 상대적인 위치로 이동이 가능하고, 스케일 바에 의한 랜덤 위치 선정도 가능하다. 내부적으로는 상대적 위치 이동에 대하여 현재 재생중인 GOP에 대한 이전 또는 이후 GOP의 첫 byte가 있는 파일 offset으로 읽는 위치를 이동함으로써 구현된다. 랜덤인 경우, 전체 파일 크기에 대비한 이동한 스케일 바의 위치를 계산하여, 그 offset의 데이터가 속한 GOP의 첫 byte부터 읽어 전송함으로써 실현되었다. 재생 속도 조절은 정상 속도의 임의의 배속으로 재생이 가능하게 하였다. 즉, 앞 또는 뒤로 가속 속도 조절 버튼을 누르면 현재 속도에 1 배 빠른 속도로 재생한다. v(>1) 속도로 재생한다면 GOP를 v-1 개씩 건너뛰어 읽고 전송함으로써 구현하였다. 속도 조절은 일반적으로 쓰여지고 있는 Real Video, Media Player 등에서는 불가능한 것으로 학생들에게 완벽한 VCR 조작 기능을 제공하는 효과를 얻는다. 또한 특징적인 기능으로써 각 클립 별로 재생의 시작 및 끝 시점을 지정할 수 있는 기능을 구현하여 반복, 집중 학습이 가능하게 하였다.

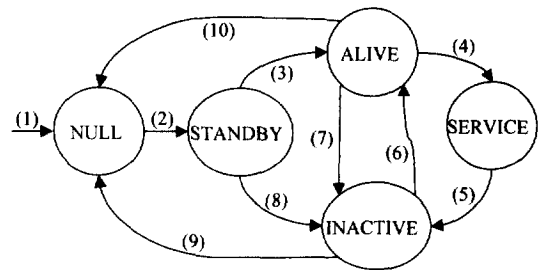
세션 종료 - 세션 종료는 크게 둘로 나뉜다. 현재 재생되고 있는 상태를 끝내고 처음 클라이언트와 서버가 세션을 맺을 때 상태로 돌아가는 종료(close)와, 다른 학습 내용을 보거나 학습을 완전히 끝내기 위해 웹 브라우저 상에서 현재 작동하고 있는 플러그인 프로그램을 종료시킴으로써 발생하는 폐쇄(teardown) 등이다. 세션의 종료 요청은 클라이언트, reader, transmitter, 그리고 서비스 게이트웨이로부터 발생할 수 있다. 클라이언트로부터의 종료 요청은 다음의 경우에 발생하는 것으로 정상적인 종료 시나리오이다. 사용자가 웹 브라우저 실행을 종료시켰을 때, 다른 웹 페이지로 옮겼을 때(이 두 경우에는 세션 폐쇄 요청도 연이어 발생함), 스트림 재생 정지(stop) 버튼을 누르거나, 웹 브라우저 윈도우 크기를 조절하였을 때, 또는 웹 페이지의 재전송(reload)을 요청했을 때 등이다. 나머지는 비정상적인 동작의 결과로서 발생한다. 여러 태스크가 제각기 세션에 대한 기능을 독립적으로 수행하고 있는 가운데 불시에 발생하는 세션 종료 신호를 처리하는 것은 매우 어려운 문제이다. 세션 종료 사유가 발생한 각 태스크에서 바로 종료 처리하게 되면 다른 태스크들이 오동작을 일으킬 수 있다. 그리고 종료 요청을 수신하는 session controller의 갑작스런 종료 작업(SCB 초기화, 버퍼 해제, 읽는 위치 초기화 등)은 다른 태스크들이 초기화되거나 해제된 자료구조를 참조하는 등의 예측할 수 없는 일을 발생시킨다. 따라서 모든 종료 요청은 항상 session controller로 우선 송신되고, 그곳에서 모든 태스크에게 종료를 알리고 각 태스크가 해당 종료 작업을 완료함을 확인한 후 최종적인 종료 작업을 session controller가 해야 한다. 이런 과정을 부드러운 종료(graceful close)라 하고, 두 단계로 이루어진다(그림 6).

(1)널(null) 세션 (2)서비스 게이트웨이의 세션 개시 요청 (3)클라이언트와 세션 오픈 (4)재생시작 (5)재생종료(stop) (6)2 단계 종료 완료 (7)클라이언트의 세션 폐쇄 요청 (8)서비스 게이트웨이의 세션 폐쇄 요청 (9)폐쇄 완료 (10)세션 종료 후 연이은 폐쇄 요청

그림 6. 세션 상태 전이도

(1) 세션 종료 요청 신호에 의해 session controller는 세션의 상태(state)를 INACTIVE로 전이시킨다. 이후, 동시에 실행되고 있는 reader와 transmitter 스레드들이 이 세션에 대한 작업을 수행할 때 현재 상태가 INACTIVE 임을 알게 되어 이 세션에 대한 서비스를 중단하는 과정을 실행한다. 만약 이 세션에서 스트림 재생이 진행 중이었다면 reader에 의해 읽혀져 버퍼에 담긴 후 아직 transmitter에 의해 미처 통신망으로 전송되지 않은 데이터 블록들이 남아 있을 수 있다. 따라서 이것을 자연스럽게 데이터 블록의 소비자인 transmitter가 전송하지 않고 바로 버퍼 풀로 되돌리도록 reader가 지시한다. 그러면 transmitter는 캐시된 데이터 블록을 전송하지 않고 버퍼 풀로 되돌리고 session controller에게 부드러운 종료의 1

단계가 완료되었음을 IPC를 통해 알린다.

(2) 1 단계 종료 작업의 완료 신호를 transmitter로부터 받은 session controller는 다음에 읽어야 하는 위치, GOP 포인터 등을 처음 세션이 성립되었을 때의 초기값으로 바꾼다. SCB의 내용(캐시 데이터 블록들의 포인터, 전송된 데이터 크기, 마지막 읽은 시간, 재생 속도, 재생 방향 등)들도 모두 초기화한다. 마지막으로 세션의 상태를 처음 세션을 맺을 때 상태인 ALIVE로 전이시킨다. 그리고 세션이 종료되었음을 클라이언트에게 알린다.

세션 폐쇄는, 세션 개시에 대한 예비작업을 하는 서비스 게이트웨이와의 교신이 성공적으로 완료된 STANDBY 상태와 세션이 클라이언트와 실제 맺어져 아직 스트림 재생이 시작되지 않은(또는 세션이 종료된) ALIVE 상태에서 완전히 세션을 없애는 제어 신호이다. 세션에 할당된 데이터 블록을 버퍼 풀로 보내고, 열려져 있는 스트림의 파일을 닫으며, 클라이언트와의 통신 채널을 없앤다. 그리고 SCB도 자유 풀로 되돌린다. 마지막으로 서비스되고 있는 스트림의 수를 하나 줄여 콘솔에 현 상황을 반영하도록 한다.

5. 여러 매체간의 공간, 시간적 동기화

멀티미디어 교실에서 활용되는 교재는 연속매체만으로 구성된 것보다는 텍스트, 그래픽, 이미지, 오디오 등이 복합적으로 한 교재 내에 포함되는 것이 일반적이다. 교재를 저작할 때 한 페이지 내에 여러 형태의 재료가 각각 어느 위치에 있는가를 규정하는 것이 공간적 동기화이다. 또한 각 재료들이 어느 시점에 재생되거나 보여져야 하는 것을 규정하는 것이 시간적 동기화이다. 이를 구현하는 방식은 크게 네가지 방식으로 구분된다. 스크립트 언어 방식, 아이콘/흐름도 방식, 시간 축 편집기 방식, 그리고 계층적 객체 방식 등이다[18]. 저작관리 시스템에서는 이를 위해 저작 용이성을 높이기 위해 여러 템플릿을 제공한다. 각 템플릿은 여러 모양의 영역으로 구분되어 있다(그림 7). 한가지 형태를 선택한 후 각 영역에 어떤 형태의 어떤 재료가 표현 내지는 재생될 것인지를 지정하고, 각 재료의 재생 또는 표현 시작, 끝 시점을 시간축을 중심으로 한 직관적인 사용자 인터페이스를 통해 설정할 수 있도록 하였다(그림 8).

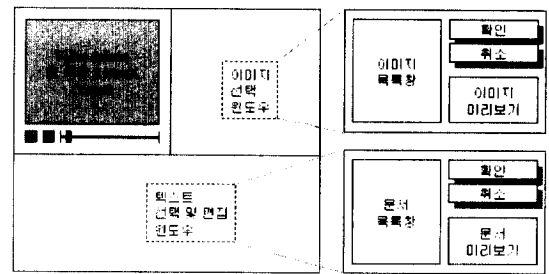



그림 7. 공간동기화 설정화면

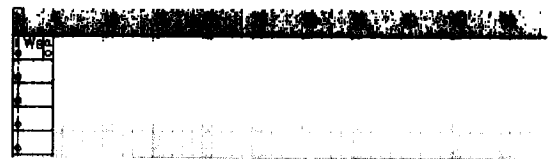



그림 8. 시간동기화 설정화면

저작된 HTML 문서에는 Java Script로 작성된 스크립트를 포함한다. 클라이언트 시스템에서는 시간동기화 정보가 포함된 Java Script에 따라 지정시간에 서비스 게이트웨이를 통한 재생 요청을 한다. 따라서 스트림 서버는 다양한 여러 매체간의 시간적 동기화에 대한 정보를 전혀 갖지 않게 됨으로써 재생 성능을 극대화하는 데 전념할 수 있었다. 구체적으로, 스크립트는 스크립팅 엔진(scripting engine)이라는 별도의 내부 프로세스 서버(In-Process Server)로 구현된 COM 컴포넌트에 의해 해석되고 실행된다. 이와 함께 웹 브라우저는 ActiveX 컨트롤의 컨테이너로서의 역할도 수행하므로, HTML 문서를 로드할 때 ActiveX 컨트롤도 함께 로드한다. 스크립팅 엔진이 HTML 문서에 포함된 스크립트를 해석하여 실행할 때 웹 브라우저에 내장된 객체뿐만 아니라 함께 로드된 ActiveX 컨트롤과도 상호작용을 할 수 있는 기능을 활용하여 매체간의 시간적 동기화를 실현하였다.

6. 실행 예

클라이언트는 윈도우 95 시스템에서 웹을 기반으로 서비스 게이트웨이로부터 학습 서비스를 제공받는다. 연속매체 정보 재생을 위해 별도의 MIME 형태를 정의하고 그것에 해당되는 웹 브라우저 플러그인 연속매체 재생 소프트웨어를 구현하였다. 재생은

스트림 서버와의 실시간 스트리밍 기법으로 이루어지며 VCR 조작이 가능하다. 그림 9.는 클라이언트의 실행 예를 보여준다.

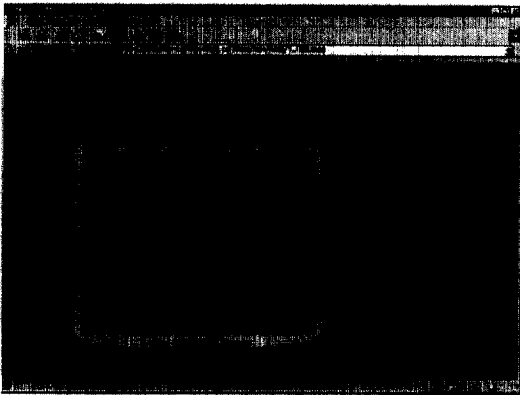



그림 9. 클라이언트 실행 예

7. 결 론

본 논문은 멀티미디어 교실을 위한 분산 멀티미디어 시스템을 이루는 구성 요소 가운데 연속매체 정보를 저장, 전송하는 스트림 서버의 구현에 관한 것이다. 기존 시스템들의 멀티미디어 교실에 적용하기에 부적합한 문제점을 극복하였다. 운영체제의 스케줄러 등을 변경함으로써 발생되는 비용으로 인한 비싼 스트림당 비용을 극복하기 위해 스트림 엔진을 응용 수준에서 구현하였다. 웹 서버 기능을 겸한 서비스 게이트웨이, 저작관리 시스템과의 유기적 연동을 통해 기존 시스템에서는 불가능한 서비스의 통합과 확장성을 확보하였다.

서버의 주요 기능을 분리하여 주기적 또는 사건(event) 위주로 실행되는 다중 스레드로 설계하였다. 서버의 성능에 결정적인 영향을 주는 디스크, 통신망 처리 스레드는 주기적으로 실행된다. 실행 시간과 간격 조절을 통해 최종 데이터의 소비자인 transmitter가 전송률을 제어함으로써 구현의 단순화 및 짧은 실행시간 복잡도를 실현하였다. 그 외 세션관리와 게이트웨이 연동 스레드는 사건 위주로 설계함으로써 context switch의 오버헤드를 최소화하였다. 현재 주기에 전송한 데이터 크기에 의거하여 다음 주기에 전송할 데이터 크기와 전송 개시 시점과 기간 등을 매 주기마다 계산하여 적용하였다. 이는 CDL과 CTL 기법을 혼합한 기법으로 다양한 재생 품질과 재생율을 가지는 여러 멀티미디어 정보를 동시에 재생하는 데 적합하다. 동시 재생 데이터가 동일한 정보일 경우 여러 학생들에게 동시에 전송할 때 IP multicasting을 DLL로 작성하여 적용하였다. 이는 방송형 수업을 하는 경우 매우 유용한 기능을 제공한다. 재생 시에 VCR 제어 기능을 GOP 단위로 가능하게 함으로써 랜덤한 실행 위치를 지정할 수 있을 뿐만 아니라 재생 속도도 조절할 수 있게 되었다. 클립 내의 특정 부분만을 반복 재생할 수 있게 지정하는 기능도 구현하여 학생들의 반복, 심화 학습을 할 수 있게 하였다. 이 기능도 기존의 시스템과 차별화되는 특징 가운데 하나이다. 세션을 열 때는 서비스 게이트웨이와의 예비 세션 개시를 통해 재생 시작 지연 시간을 크게 줄이면서 서버가 여럿 있을 경우 서버간의 부하 균할을 실현한다. 세션을 닫을 때는 여러 부분에서 발생할 수 있는 세션 종료 요청을 모두 session controller가 2 단계에 걸쳐 부드럽게 처리함으로써 능동적 태스크와의 세션 상태 동기화를 완전하게 실현할 수 있었다.

참 고 문 헌

[1] 권선만, "교육용 멀티미디어 시스템 구성 방향", 한밭교육, vol.14, pp.55~59, 대전시교육연구원, 대전, 1996

[2] Palmer W. Agnew and Anne S. Kellerman, "Distributed Multimedia," p.338, Addison Wesley, 1996

[3] Jooyoung Son, Bumjoo Seo, Kyungwook Chun, Seokjae Ha, and Yanghee Choi, "Architectural Design of a Large-scale Multimedia Information Retrieval Service," Proc of Pacific Workshop on Distributed Multimedia Systems(DMS) '97, pp. 17~24, Vancouver, July 1997.

[4] J. Beveridge, R. Wiener, "Multithreading Applications in Win32", Addison Wesley, 1997.

[5] P. V. Rangan, H. M. Vin, and S. Ramanathan, "Designing an On-Demand Multimedia Service," IEEE Communications Magazine, Vol. 30, No. 7, pp. 56~64, July 1992.

[6] D. P. Anderson, Y. Osawa and R. Govindan,

"Real-Time Disk Storage and Retrieval of Digital Audio and Video," Technical Report No. UCB/CSD 91/646, Computer Science Division, University of California, Berkeley, Aug. 1991.
[7] J. Gemmell and S. Christodoulakis, "Principles of Delay-Sensitive Multimedia Data Storage and Retrieval," ACM Transactions on Information Systems, Vol. 10, No.1, pp. 51~90, Jan. 1992.
[8] Jooyoung Son and Yanghee Choi, "An Efficient Storage Scheme for Multimedia Server," IEICE Transactions on Information and Systems, Special Issue on Multimedia Computing and Communications, Vol.E79-D, No.6, pp.712~718, June 1996.
[9] http://msdn.microsoft.com/library/sdkdoc/winsock "IP multicast," Microsoft, 1998.
[10] 손주영, "개인 멀티미디어 서비스:멀티미디어 정보통신 서비스 모델과 기술에 대한 새로운 접근", 한국해양대 산업기술연구소 연구논문집 Vol. 16, pp.163~179, 1998.
[11] LG전자, "LG 네오비스 비디오 서버 소개", LG전자 미디어통신연구소 기술문서, 1998.
[12] Jose Alvear, "Web Developer.Com Guide to Streaming Multimedia," John Wiley & Sons, 1998.
[13] LG전자, "Solaris 2.5.1 기반 네오비스 비디오 서버 개발 완료 보고서", LG전자 미디어통신연구소 기술문서, 1998.
[14] E. Chang and A. Zakhor, "Cost Analysis for VBR Video Servers," Proc. of MMCN '95, San Jose, USA, 1995.
[15] ISO/IEC 11172, "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s," 1993.
[16] RFC 1889, "RTP: A Transport Protocol for Real-Time Applications," IETF, 1996.
[17] 손종문, 김길용, 김해진, "범용 운영 체제상에서 비디오 스트림 서비스를 위한 실시간 스케줄링", 정보과학회 논문지 제3권 제5호 pp.498~509, 1997.
[18] 김명호, 이윤준, "멀티미디어 개념 및 응용", pp.259~278, 홍릉과학출판사, 1997.

손 주 영

1981년~1985년 서울대학교 계산통계학과 학사.
1991년~1993년 서울대학교 컴퓨터공학과 석사
1993년~1997년 서울대학교 컴퓨터공학과 박사.
1985년~1987년 금성반도체(주) 연구원
1987년~1998년 LG전자(주) 책임연구원
1998년~현재 한국해양대학교 자동화정보공학부 전임강사.
관심분야 : 멀티미디어 통신 및 시스템, 초고속통신망 프로토콜, 이동 단말의 인터넷 서비스 구조 및 프로토콜